THE XTAG PROJECT AT PENN*

Aravind K. Joshi
Department of Computer and Information Science and
Institute for Research in Cognitive Science
University of Pennsylvania
Philadelphia, PA
USA
joshi@linc.cis.upenn.edu

Abstract

The XTAG project at the University of Pennsylvania has been an on-going project since about 1988. At present, the project consists of (1) the construction of a wide coverage lexicalized tree-adjoining grammar (LTAG) for English, relatively smaller grammars for Chinese, Korean, and Hindi, and the associated parsers, including statistical processing and (2) extraction of LTAG grammars from annotated corpora and their use in statistical parsing, for improving annotations, and for cross-linguistic mappings useful for machine translation. These two main directions of the project provide a unique environment for pursuing several formal, linguistic, computational, and statistical aspects of natural language processing. In this paper we will present an introduction to LTAG, an overview of the XTAG project, and a brief description of some specific efforts, especially those related to parsing.

1 Introduction

The XTAG project at the University of Pennsylvania has been an on-going project since about 1988. At present, the project consists of (1) the construction of a wide coverage lexicalized tree-adjoining grammar (LTAG) for English[XTAG Research Group, 2001] and relatively smaller grammars for Chinese, Korean, and Hindi, and the associated parsers, including statistical processing and (2) extraction of LTAG grammars from annotated corpora and their use in statistical parsing, for improving annotations, and for cross-linguistic mappings useful for machine translation. These two main directions of the project provide a unique environment for pursuing several formal, linguistic, computational, and statistical aspects of natural language processing. In this paper we will present an introduction to LTAG, an overview of the XTAG project, and a brief description of some specific efforts, especially those related to parsing. The plan of the paper is as follows. In Section 2 TAGs (LTAGs) are introduced, providing a discussion of the important issue of lexicalization and how it leads to TAGs (LTAGs). In Section 3 some important properties of TAGs have been described. In Section 4 some selected efforts (especially some recent ones) in the XTAG project have been discussed briefly, followed by a conclusion section (Section 5).

The earliest stochastic variants of TAG were proposed by [Resnik, 1992, Schabes, 1992]. LTAG grammars have been extracted from annotated corpora[Xia et al., 2001, Xia, 2001, Chiang, 2000][1], which in turn have been used for statistical parsing [Chiang, 2000, Sarkar, 2001]. The statistical

* This work was partially supported by NSF Grant SBR8920230. I want to thank David Chiang, Carlos Prolo, Anoop Sarkar, and Fei Xia for providing valuable help for the preparation of this paper.

[1]See also [Chen and Vijay-Shanker, 2000], which is not a part of the XTAG project. This work is being carried out at the University of Delaware.

parsing work done in TAGs emphasizes the use of lexicalized elementary trees and the recovery of the best derivation for a given sentence rather than the best parse tree.

2 Tree-adjoining grammars

Tree-adjoining grammar (TAG) is a formal tree rewriting system. TAG and Lexicalized Tree-Adjoining Grammar (LTAG) have been extensively studied both with respect to their formal properties and to their linguistic relevance. TAG and LTAG are formally equivalent, however, from the linguistic perspective LTAG is the system we will be concerned with in this paper. We will often use these terms TAG and LTAG interchangeably.

The motivations for the study of LTAG are both linguistic and formal. The elementary objects manipulated by LTAG are structured objects (trees or directed acyclic graphs) and not strings. Using structured objects as the elementary objects of the formal system, it is possible to construct formalisms whose properties relate directly to the study of strong generative capacity (i.e., structural descriptions), which is more relevant to the linguistic descriptions than the weak generative capacity (sets of strings).

Each grammar formalism specifies a domain of locality, i.e., a domain over which various dependencies (syntactic and semantic) can be specified. It turns out that the various properties of a formalism (syntactic, semantic, computational, and even psycholinguistic) follow, to a large extent, from the initial specification of the domain of locality.

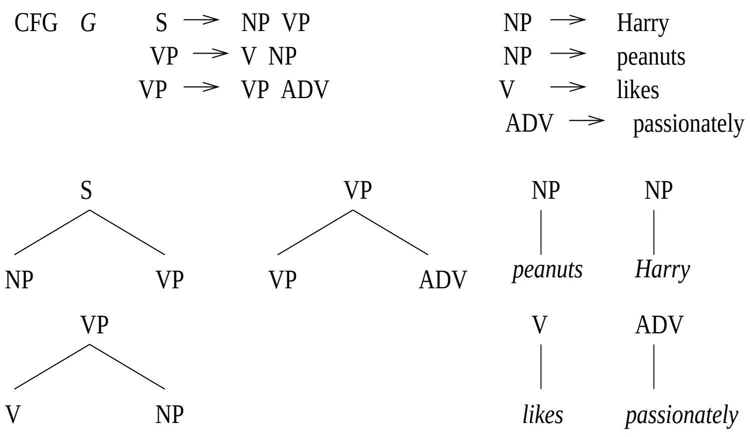

Figure 1: Domain of locality of a context-free grammar

2.1 Domain of locality of CFGs

In a context-free grammar (CFG) the domain of locality is the one level tree corresponding to a rule in a CFG (Fig. 1). It is easily seen that the arguments of a predicate (for example, the two arguments of *likes*) are not in the same local domain. The two arguments are distributed over the two rules (two domains of locality)– $S \rightarrow NP\ VP$ and $VP \rightarrow V\ NP$. They can be brought together by introducing a rule $S \rightarrow NP\ V\ VP$. However, then the structure provided by the VP node is lost. We should also note here that not every rule (domain) in the CFG in (Fig. 1) is lexicalized. The four rules on the right are lexicalized, i.e., they have a lexical anchor. The rules on the left are not lexicalized. The second and the third rules on the left are almost lexicalized, in the sense that they each have a preterminal category (V in the second rule and ADV in the third rule), i.e., by replacing V by *likes* and ADV by *passionately* these two rules will become lexicalized. However, the first rule on the left ($S \rightarrow NP\ VP$) cannot be lexicalized. Can a CFG be lexicalized, i.e., given a CFG, G, can we construct another CFG, G', such that every rule in G' is lexicalized and $T(G)$, the set of (sentential) trees (i.e., the tree language of G) is the same as the tree language $T(G')$ of G'? It can be shown that this is not the case [Joshi and Schabes, 1997]. Of course, if we require that only the string languages of G and G' be the same (i.e., they are weakly equivalent) then any CFG can be lexicalized. This follows from the fact that any CFG can be put in the Greibach normal form where each rule is of the form $A \rightarrow w\ B1\ B2\ ...\ Bn$ where w is a lexical item and the $B's$ are nonterminals. The lexicalization we are interested in requires the tree languages (i.e., the set of structural descriptions) be the same, i.e., we are interested in the bf 'strong' lexicalization. To summarize, a CFG cannot be strongly lexicalized by a CFG. This follows from the fact that the domain of locality of CFG is a one level tree corresponding to a rule in the grammar. Note that there are two issues we are concerned with here– lexicalization of each elementary domain and the encapsulation of the arguments of the lexical anchor in the elementary domain of locality. The second issue is independent of the first issue. From the mathematical point of view the first issue, i.e., the lexicalization of the elementary domains of locality is the crucial one. We can obtain strong lexicalization without satisfying the requirement specified in the second issue (encapsulation of the arguments of the lexical anchor). Of course, from the linguistic point of view the second issue is very crucial. What this means is that among all possible strong lexicalizations we should choose only those that meet the requirements of the second issue. For our discussions in this paper we will assume that we always make such a choice.

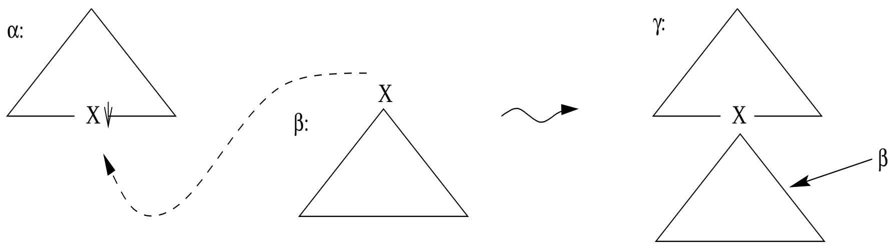

Figure 2: Substitution

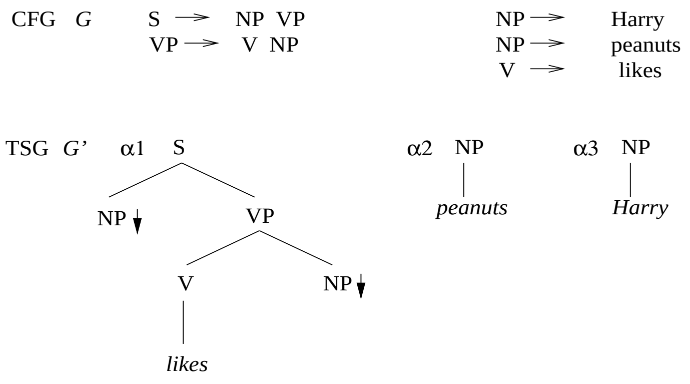

Figure 3: Tree substitution grammar

2.2 Lexicalization of CFGs

Now we can ask the following question. Can we strongly lexicalize a CFG by a grammar with a larger domain of locality? Fig. 2 and Fig. 3 show a tree substitution grammar where the elementary objects (building blocks) are the three trees in Fig. 3 and the combining operation is the tree substitution operation shown in Fig. 2. Note that each tree in the tree substitution grammar (TSG), G' is lexicalized, i.e., it has a lexical anchor. It is easily seen that G' indeed strongly lexicalizes G. However, TSGs fail to strongly lexicalize CFGs in general. We show this by an example. Consider the CFG, G, in Fig. 4 and a proposed TSG, G'. It is easily seen that although G and G' are weakly equivalent they are not strongly equivalent. In G', suppose we start with the tree α_1 then by repeated substitutions of trees in G' (a node marked with a vertical arrow denotes a substitution site) we can grow the right side of α_1 as much as we want but we cannot grow the left side. Similarly for α_2 we can grow the left side as much as we want but not the right side. However, trees in G can grow on both sides. Hence, the TSG, G', cannot strongly lexicalize the CFG, G [Joshi and Schabes, 1997].

CFG G S $\rightarrow$ S S (non-lexical)
S $\rightarrow$ a (lexical)

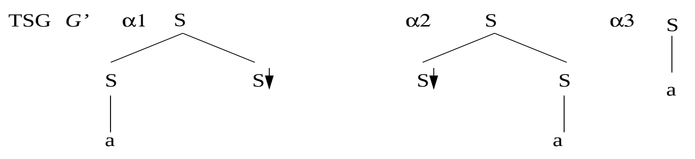

Figure 4: A tree substitution grammar

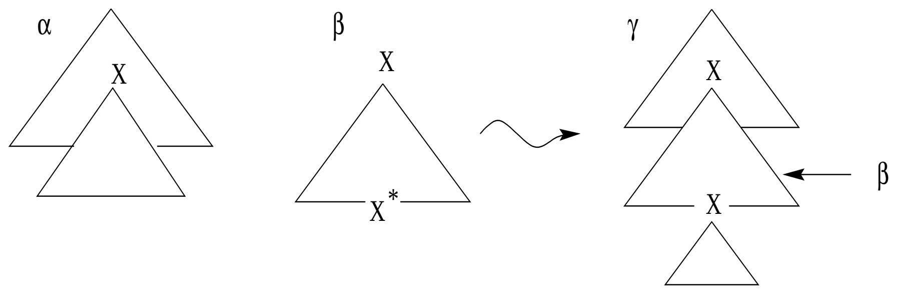

Figure 5: Adjoining

We now introduce a new operation called 'adjoining' as shown in Fig. 5. Adjoining involves splicing (inserting) one tree into another. More specifically, a tree β as shown in Fig. 5 is inserted (adjoined) into the tree α at the node X resulting in the tree γ. The tree β, called an **auxiliary tree**, has a special form. The root node is labeled with a nonterminal, say X and on the frontier there is also a node labeled X called the foot node (marked with *). There could be other nodes (terminal or nonterminal) nodes on the frontier of β, the nonterminal nodes will be marked as substitution sites (with a vertical arrow). Thus if there is another occurrence of X (other than the foot node marked with *) on the frontier of β it will be marked with the vertical arrow and that will be a substitution site. Given this specification, adjoining β to α at the node X in α is uniquely defined. Adjoining can

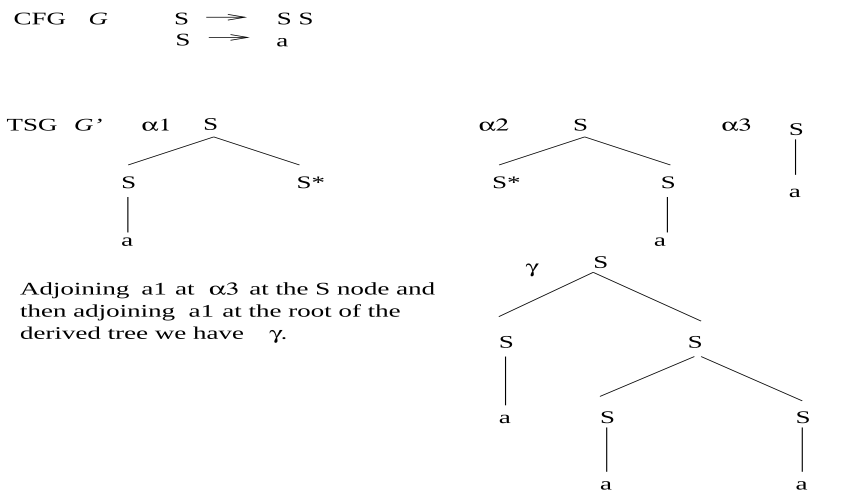
CFG G
S → S S
S → a
TSG G'
α1
α2
α3
S
S*
a
γ
Adjoining a1 at α3 at the S node and then adjoining a1 at the root of the derived tree we have γ.

Figure 6: Adjoining arises out of lexicalization

also be seen as a pair of substitutions as follows: The subtree at X in α is detached, β is substituted at X and the detached subtree is then substituted at the foot node of β. A tree substitution grammar when augmented with the adjoining operation is called a tree-adjoining grammar (lexicalized tree-adjoining grammar because each elementary tree is lexically anchored). In short, LTAG consists of a finite set of elementary trees, each lexicalized with at least one lexical anchor. The elementary trees are either initial or auxiliary trees. Auxiliary trees have been defined already. Initial trees are those for which all nonterminal nodes on the frontier are substitution nodes. It can be shown that any CFG can be strongly lexicalized by an LTAG [Joshi and Schabes, 1997].

In Fig. 6 we show a TSG, G', augmented by the operation of adjoining, which strongly lexicalizes the CFG, G. Note that the LTAG looks the same as the TSG considered in Fig. 4. However, now trees α_1 and α_2 are auxiliary trees (marked with *) that can participate in adjoining. Since adjoining can insert a tree in the interior of another tree it is possible to grow both sides of the tree α_1 and tree α_2, which was not possible earlier with substitution alone. In summary, we have shown that by increasing the domain of locality we have achieved the following: (1) lexicalized each elementary domain, (2) introduced an operation of adjoining, which would not be possible without the increased domain of locality (note that with one level trees as elementary domains adjoining becomes the same as substitution since there are no interior nodes to be operated upon), and (3) achieved strong lexicalization of CFGs.

2.3 Lexicalized tree-adjoining grammars

Rather than giving formal definitions for LTAG and derivations in LTAG we will give a simple example to illustrate some key aspects of LTAG. We show some elementary trees of a toy LTAG grammar of English. Fig. 7 shows two elementary trees for a verb such as *likes*. The tree α_1 is anchored on *likes* and encapsulates the two arguments of the verb. The tree α_2 corresponds to the object extraction construction. Since we need to encapsulate all the arguments of the verb in each elementary tree for *likes*, for the object extraction construction, for example, we need to make the elementary tree associated with *likes* large enough so that the extracted argument is in the same elementary domain.

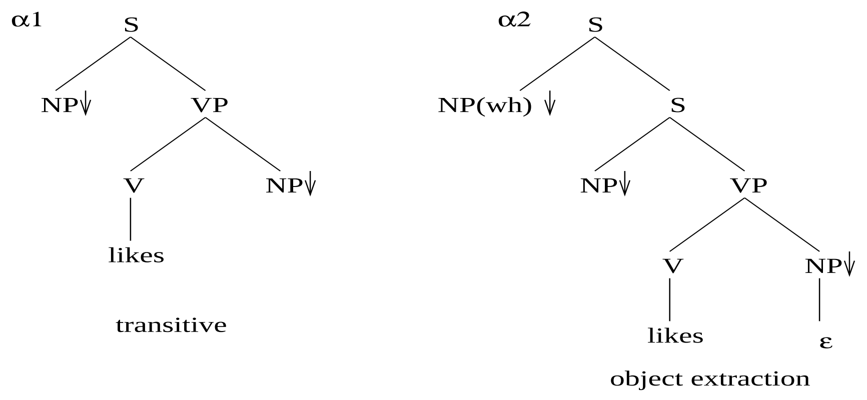

Figure 7: LTAG: Elementary trees for *likes*

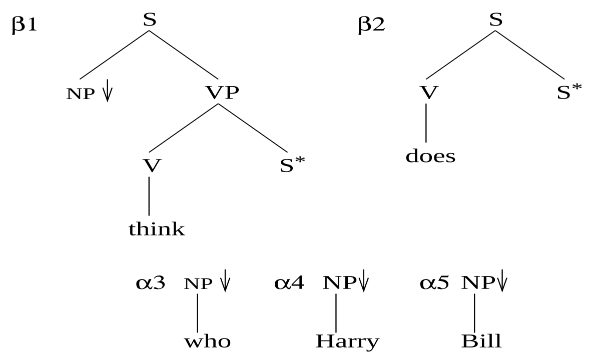

Figure 8: LTAG: Sample elementary trees

Thus, in principle, for each 'minimal' construction in which *likes* can appear (for example, subject extraction, topicalization, subject relative, object relative, passive, etc.) there will be an elementary tree associated with that construction. By 'minimal' we mean when all recursion has been factored away. This factoring of recursion away from the domain over which the dependencies have to be specified is a crucial aspect of LTAGs as they are used in linguistic descriptions. This factoring allows all dependencies to be localized in the elementary domains. In this sense, there will, therefore, be no long distance dependencies as such. They will all be local and will become long distance on account of the composition operations, especially adjoining.

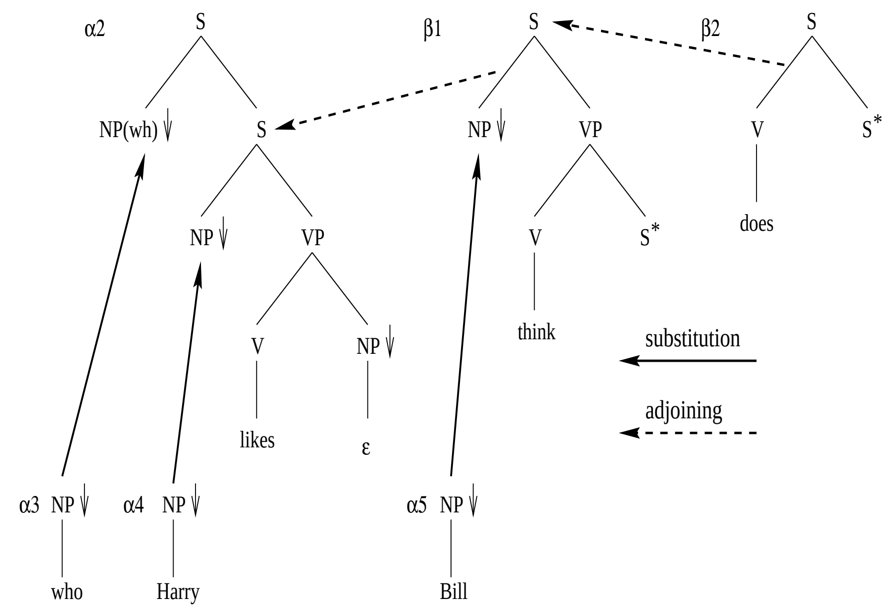

Figure 9: LTAG derivation for *who does Bill think Harry likes*

Fig. 8 shows some additional trees. Trees α_3, α_4, and α_5 are initial trees and trees β_1 and β_2 are auxiliary trees with foot nodes marked with *. A derivation using the trees in Fig. 7 and Fig. 8 is shown in Fig. 9. The trees for *who* and *Harry* are substituted in the tree for *likes* at the respective NP nodes, the tree for *Bill* is substituted in the tree for *think* at the NP node, the tree for *does* is adjoined to the root node of the tree for *think* tree (adjoining at the root node is a special case of adjoining), and finally the derived auxiliary tree (after adjoining β_2 to β_1) is adjoined to the indicated interior S node of the tree α_2. This derivation results in the **derived tree** for *who does Bill think Harry likes* as shown in Fig. 10. Note that the dependency between *who* and the complement NP in α_2 (local to that tree) has been stretched in the derived tree in Fig. 10. This tree is the conventional tree associated with the sentence.

However, in LTAG there is also a derivation tree, the tree that records the history of composition of the elementary trees associated with the lexical items in the sentence. This derivation tree is shown in Fig. 11. The nodes of the tree are labeled by the tree labels such as α_2 together with the lexical anchor.[2] The derivation tree is the crucial derivation structure for LTAG. We can obviously build the derived tree from the derivation tree. For semantic computation the derivation tree (and not the derived tree)

[2]The derivation trees of LTAG have a close relationship to the dependency trees, although there are some crucial differences; however, the semantic dependencies are the same. See [Rambow and Joshi, 1995]

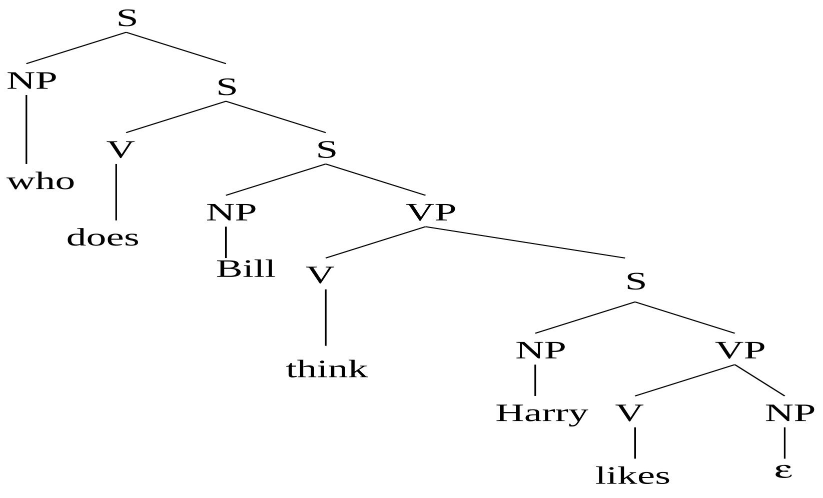

Figure 10: LTAG derived tree for *who does Bill think Harry likes*

is the crucial object. Compositional semantics is defined on the derivation tree. The idea is that for each elementary tree there is a semantic representation associated with it and these representations are composed using the derivation tree. Since the semantic representation for each elementary tree is directly associated with the tree there is no need to reproduce necessarily the internal hierarchy in the elementary tree in the semantic representation [Joshi and Vijay-Shanker, 1999]. This allows the so-called 'flat' semantic representation as well as helps in dealing with some non-compositional aspects as in the case of rigid and flexible idioms.

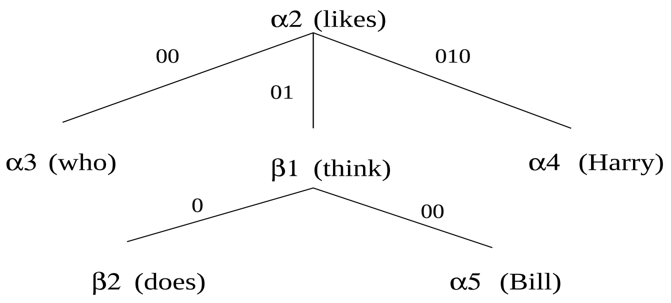

Figure 11: LTAG derivation tree

3 Some important properties of LTAG

The two key properties of LTAG are (1) extended domain of locality (EDL) (for example, as compared to CFG), which allows (2) factoring recursion from the domain of dependencies (FRD), thus making all dependencies local. All other properties of LTAG (mathematical, linguistic, and even psycholinguistic) follow from EDL and FRD. TAGs (LTAGs) belong to the so-called class of mildly context-sensitive grammars [Joshi, 1985]. Context-free languages (CFL) are properly contained in the class of languages of LTAG, which in turn are properly contained in the class of context-sensitive languages. There is a machine characterization of TAG (LTAG), called embedded pushdown automaton

(EPDA) [Vijay-Shanker, 1987],i.e., for every TAG language there is an EPDA which corresponds to this (and only this) language and the language accepted by any EPDA is a TAG language. EPDAs have been used to model some psycholinguistic phenomena, for example, processing crossed dependencies and nested dependencies have been discussed in [Joshi, 1990]. With respect to formal properties, the class of TAG languages enjoys all the important properties of CFLs, including polynomial parsing (with complexity $O(n^6)$).

Large scale wide coverage grammars have been built using LTAG, the XTAG system (LTAG grammar and lexicon for English and a parser) being the largest so far (for further details see [XTAG Research Group, 2001]. In the XTAG system, each node in each LTAG tree is decorated with two feature structures (top and bottom feature structures), in contrast to the CFG based feature structure grammars. This is necessary because adjoining can augment a tree internally, while in a CFG based grammar a tree can be augmented only at the frontier. It is possible to define adjoining and substitution (as it is done in the XTAG system) in terms of appropriate unifications of the top and bottom feature structures. Because of FRD (factoring recursion from the domain of dependencies), there is no recursion in the feature structures. Therefore, in principle, feature structures can be eliminated. However, they are crucial for linguistic descriptions. Constraints on substitution and adjoining are modeled via these feature structures [Vijay-Shanker, 1987]. This method of manipulating feature structures is a direct consequence of the extended domain of locality of LTAG.

4 Stochastic TAGs (LTAGs) and extracted LTAGs

What is the relevance of LTAGs to statistical parsing? It might be thought that its added formal power makes parameter estimation unnecessarily difficult; or that whatever benefits it provides—the ability to model unbounded cross-serial dependencies, for example—are inconsequential for statistical parsing, which is concerned with the probable rather than the possible.

However, just as TAG is not, by itself, a complete linguistic theory, but a formalism for specifying linguistic theories, it should not be viewed as a statistical model but a formalism for specifying statistical models. The advantage that TAG has over CFG is that it assigns richer *structural descriptions* to sentences; specifically, in addition to parse trees, it assigns *derivation trees* (see Section 2) on which features of a parsing model can be defined.

[Chiang, 2001, Chiang, 2000] gives a statistical parser based on stochastic Tree Insertion Grammars, a variant of TAGs introduced in [Schabes and Waters, 1994], which constrains the operation of adjoining in way such that the weak generative power is equivalent to CFGs but the strong generative power (relevant to structural descriptions) is strictly greater than that for CFGs. The experiments were based on fully lexicalized elementary trees and achieves 87.6% labeled precision and 87.4% labeled recall. These results show that one does not have to sacrifice performance over lexicalized PCFGs while maintaining a more elaborate model using TAGs. [Chiang, 2001] also reports results on the Chinese Treebank[3]. This involved only minor changes to the English parser.

[3]The Chinese Treebank is not part of the XTAG project. It is a separate project. The work is being carried out by Fei Xia. The project is directed by Martha Palmer. The goal of the project is to build a large-scale high-quality Treebank for Chinese. The first portion of the Treebank, which has about 100 thousand words, was released to the public in 2000. Since then, more data from various sources have been annotated. The first portion of the Treebank consists of 325 articles from the Xinhua newswire published in 1994-1998 (the majority of these documents focus on economic developments while the rest describe general political and cultural topics). It contains 172 thousand *hanzi* (Chinese characters), or 100 thousand words after word segmentation.

Chiang's use of a variant of probabilistic TAG captures the same bilexical dependencies that these PCFG-based models do (a possibility noted early on by [Resnik, 1992, Schabes, 1992], but with less notational overhead, and demonstrate that it can be used to parse with comparable accuracy. He argues that the use of probabilistic TAG provides two benefits over PCFG: first, it naturally captures dependencies that must be encoded *ad hoc* into a PCFG, including dependencies which the PCFG-based parsers do not capture; second, the derivation trees a TAG parser computes in addition to parse trees are useful for further processing (for example, translation or semantic interpretation)[4]

[Xia et al., 2001, Xia, 2001] reports on an algorithm (LexTract) that permit the extraction of TAG derivation trees from Treebanks in various languages. The algorithm uses only minimal edits to tables of data that are localized to each new Treebank. The extraction process has three steps: (1) LexTract fully brackets each tree in the Penn Treebank. This is because in the Penn Treebank 'modifier' structures, in general, are shown as flat structures. (2) LexTract decomposes the fully bracketed trees into a set of elementary trees of LTAG. (3) LexTract builds the derivation tree for the fully bracketed trees. Xia also makes a comparison of the extracted grammar with the XTAG grammar and a cross-linguistic study of extracted grammars for English, Chinese, and Korean [Xia et al., 2001].

[Sarkar, 2001] explores some new machine learning techniques to enable statistical parsers to take advantage of unlabeled data. By exploiting the representation of stochastic TAG to view parsing as a classification task, it uses a machine learning method called Co-Training to iteratively label new training data for the parser, improving its performance over simply using the available amount of labeled data. While training only on the labeled set gave a performance of 72.23% and 69.1% labeled bracketing precision and recall, the technique achieves 80.02% and 79.64% labeled bracketing precision and recall using Co-Training with a labeled set of about 10K sentences and an unlabeled set of 30K sentences. This is preliminary work and experiments are in progress over large datasets.

[Srinivas, 1997b, Srinivas, 1997a] describes a method of partial parsing that uses local attachment heuristics after a probabilistic method that picks the best elementary tree for each word in a sentence: a technique termed as SuperTagging indicating the affinity between the problems of assigning complex structures such as trees to each word in a sentence as compared to the assignment of part of speech tags.

Prolo is currently developing a large scale LR parser for LTAG aiming to produce a practical LR parser by relaxing some of the LR theoretical assumptions. He uses corpus-based statistical techniques to resolve parsing conflicts [Prolo, 2000]. The parser is being tested on TAG grammars extracted from the English Penn Treebank.

5 Conclusion

We have given an introduction to LTAG, emphasizing especially the role of lexicalization and how LTAGs arise from CFGs in the process of lexicalization. This aspect of LTAGs is highly relevant to parsing. We have then provided an overview of the XTAG project. The two main aspects of the XTAG project (a) the construction of wide coverage grammars and the extraction of the grammars from

[4]The following work by Hwa [Hwa, 1998], Harvard Univerity, is not part of the XTAG project. We mention it here because it is highly relevant to the work on stochastic TAG. Hwa uses the inside-outside algorithm for stochastic Tree Insertion Grammars defined in [Schabes, 1992] and combines this with the use of inside-outside training from partially bracketed Treebank data [Pereira and Schabes, 1992]. The experiments reported were conducted on the WSJ Penn Treebank with the input to the learning algorithm being part-of-speech tags (rather than the words themselves). [Hwa, 1999] extends the partial bracketing approach by suppressing various kinds of labeled brackets as a possible way of minimizing annotation cost by recovering some labeled brackets automatically.

annotated corpora and (b) the construction of the associated parsers, provide a unique environment for pursuing several formal, linguistic, computational, and statistical aspects of natural language processing. We have also described some specific efforts, especially those related to parsing.

References

[Chen and Vijay-Shanker, 2000] Chen, J. and Vijay-Shanker, K. (2000). Automated Extraction of TAGs from the Penn Treebank. In *Proc. of the 6th International Workshop on Parsing Technologies (IWPT-2000), Italy.*

[Chiang, 2000] Chiang, D. (2000). Statistical Parsing with an Automatically-Extracted Tree Adjoining Grammar. In *Proc. of ACL-2000.*

[Chiang, 2001] Chiang, D. (2001). Statistical parsing with tree adjoining grammars. In *Data Oriented Parsing*. CSLI. In this volume.

[Hwa, 1998] Hwa, R. (1998). Learning Parse and Translation Decisions from Examples with Rich Context. In *Proceedings of COLING-ACL '98*, pages 557–563, Montreal.

[Hwa, 1999] Hwa, R. (1999). Supervised Grammar Induction Using Training Data with Limited Constituent Information. In *Proceedings of 37th Meeting of the ACL (ACL '99)*, pages 20–26, College Park, MD.

[Joshi and Schabes, 1997] Joshi, A. and Schabes, Y. (1997). Tree adjoining grammars. In Rozenberg, G. and Salomaa, A., editors, *Handbook of Formal Languages and Automata*. Springer-Verlag.

[Joshi and Vijay-Shanker, 1999] Joshi, A. and Vijay-Shanker, K. (1999). Compositional Semantics with LTAG: How Much Underspecification Is Necessary? In *Proc. of 3nd International Workshop on Computational Semantics.*

[Joshi, 1985] Joshi, A. K. (1985). Tree Adjoining Grammars: How much context Sensitivity is required to provide a reasonable structural description. In Dowty, D., Karttunen, I., and Zwicky, A., editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge, U.K.

[Joshi, 1990] Joshi, A. K. (1990). Processing crossed and nested dependencies: an automaton perspective on psycholinguisitc results. *Language and Cognitive Processes*, 5(1).

[Pereira and Schabes, 1992] Pereira, F. and Schabes, Y. (1992). Inside-Outside Reestimation from Partially Bracketed Corpora. In *Proceedings of 30th Meeting of the ACL (ACL '92)*, pages 128–135.

[Prolo, 2000] Prolo, C. A. (2000). An efficient LR parser generator for Tree Adjoining Grammars. In *Proceedings of the 6th International Workshop on Parsing Technologies (IWPT-2000)*, Trento, Italy.

[Rambow and Joshi, 1995] Rambow, O. and Joshi, A. (1995). A formal look at dependency grammars and phrase-structure grammars, with special consideration of word-order phenomena. In Wanner, L., editor, *Current Issues in Meaning-Text Theory*. Pinter, London.

[Resnik, 1992] Resnik, P. (1992). Probabilistic tree-adjoining grammars as a framework for statistical natural language processing. In *Proc. of COLING '92*, volume 2, pages 418–424, Nantes, France.

[Sarkar, 2001] Sarkar, A. (2001). Applying co-training methods to statistical parsing. In *Proceedings of NAACL 2001*, Pittsburgh, PA.

[Schabes, 1992] Schabes, Y. (1992). Stochastic lexicalized tree-adjoining grammars. In *Proc. of COLING '92*, volume 2, pages 426–432, Nantes, France.

[Schabes and Waters, 1994] Schabes, Y. and Waters, R. (1994). Tree insertion grammars. Technical Report TR-94-13, Mitsubishi Electric Research Laboratories.

[Srinivas, 1997a] Srinivas, B. (1997a). *Complexity of Lexical Descriptions and its Relevance to Partial Parsing.* PhD thesis, Department of Computer and Information Sciences, University of Pennsylvania.

[Srinivas, 1997b] Srinivas, B. (1997b). Performance Evaluation of Supertagging for Partial Parsing. In *Proceedings of Fifth International Workshop on Parsing Technology*, Boston, USA.

[Vijay-Shanker, 1987] Vijay-Shanker, K. (1987). *A Study of Tree Adjoining Grammars.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

[Xia, 2001] Xia, F. (2001). *Investigating the Relationship between Grammars and Treebanks for Natural languages.* PhD thesis, University of Pennsylvania, Philadelphia, PA.

[Xia et al., 2001] Xia, F., Han, C., Palmer, M., and Joshi, A. (2001). Automatically Extracting and Comparing Lexicalized Grammars for Different Languages. In *Proc. of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, Seattle, Washington.

[XTAG Research Group, 2001] XTAG Research Group (2001). A lexicalized tree adjoining grammar for english. Technical Report IRCS-01-03, IRCS, University of Pennsylvania.